How to Compile A Quantum Bayesian Net

Robert R. Tucci
P.O. Box 226
Bedford, MA 01730
tucci@ar-tiste.com

February 1, 2008

Abstract

We show how to express the information contained in a Quantum Bayesian (QB) net as a product of unitary matrices. If each of these unitary matrices is expressed as a sequence of elementary operations (operations such as controlled-nots and qubit rotations), then the result is a sequence of operations that can be used to run a quantum computer. QB nets have been run entirely on a classical computer, but one expects them to run faster on a quantum computer.

Introduction

Quantum Bayesian (QB) Nets[1]-[3] are a method of modelling quantum systems graphically in terms of network diagrams. In this paper, we show how to express the information contained in a QB net as a product of unitary operators. In another paper[4], we have presented a method for reducing any unitary operator to a sequence of elementary operations (SEO). Such SEOs can be used to manipulate an array of quantum bits, a quantum computer.[5]-[7] Thus, combining the results of this paper and those of Ref.[4], one can reduce a QB net into a SEO that can be used to run a quantum computer. Of course, QB nets can and have been run entirely on a classical computer[3]. However, because of the higher speeds promised by quantum parallelism, one expects QB nets to run much faster on a quantum computer.

The process of reducing a QB net to a SEO is analogous to the process of "compiling source code" for classical computers. With classical computers, one writes a computer program in a high level language (like Fortran, C or C++). A compiler then expresses this as a SEO for manipulating bits. In the case of quantum computers, a QB net may be thought of as a program in a high level language. In the future, programs called "quantum compilers" will be widely available. These will run (initially) on classical computers. They will take whatever QB net we give it, and re-express it as a SEO that will then be used to control an array of quantum bits. These quantum compilers will take the QB net entered by the user and automatically add to it quantum error correction code[7] and various speed optimizations.

QB nets are to quantum physics what Classical Bayesian (CB) nets[8] are to classical physics. CB nets have been used very successfully in the field of artificial intelligence (AI). In fact, even Microsoft's Office Suite contains CB net code[9]. Thus, we hope and expect that some day QB nets, running on quantum computers, will be used for AI applications. In fact, we believe that quantum computers are ideally suited for such applications. First, because AI tasks often require tremendous power, and quantum computers seem to promise this. Second, because quantum computers are plagued by quantum noise, which makes their coherence times short. There are palliatives to this, such as quantum error correction[7]. But such palliatives come at a price: a large increase in the number of steps. The current literature often mentions factoring a large number into primes[6] as a future use of quantum computers. However, due to noise, quantum computers may ultimately prove to be impractical for doing long precise calculations such as this. On the other hand, short coherence times appear to be a less serious problem for the types of calculations involved in AI. The human brain has coherence times too short to factor a 100 digit number into primes, and yet long enough to conceive the frescoes in the Sistine Chapel. We do not mean to imply that the human brain is a quantum computer. An airplane is not a bird, but it makes a good flyer. Perhaps a quantum computer, although not a human brain, can make a good thinker.

Review

We begin by presenting a brief review of QB nets. For more information, see Ref.[1]-[3].

In what follows, we use the following notation. We define $Z_{a,b} = \{a, a+1, \ldots, b\}$ for any integers a and b. $\delta(x, y)$ equals one if $x = y$ and zero otherwise. For any finite set S, $|S|$ denotes the number of elements in S.

We call a *graph* (or a diagram) a collection of nodes with arrows connecting some pairs of these nodes. The arrows of the graph must satisfy certain constraints. We call a *labelled graph* a graph whose nodes are labelled. A *QB net* consists of two parts: a labelled graph with each node labelled by a random variable, and a collection of node matrices, one matrix for each node. These two parts must satisfy certain constraints.

An *internal arrow* is an arrow that has a starting (source) node and a different ending (destination) one. We will use only internal arrows. We define two types of nodes: an *internal node* is a node that has one or more internal arrows leaving it, and an *external node* is a node that has no internal arrows leaving it. It is also common to use the terms *root node* or *prior probability node* for a node which has no incoming arrows (if any arrows touch it, they are outgoing ones).

We restrict our attention to *acyclic* graphs; that is, graphs that do not contain cycles. (A *cycle* is a closed path of arrows with the arrows all pointing in the same sense.)

We assign a random variable to each node of the QB net. (Henceforth, we will underline random variables. For example, we might write $P(\underline{x} = x)$ for the probability that the random variable $\underline{x}$ assumes the particular value x.) Suppose the random variables assigned to the N nodes are $\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_N$. For each $j \in Z_{1,N}$, the random variable $\underline{x}_j$ will be assumed to take on values within a finite set Σ_j called *the set of possible states of* $\underline{x}_j$.

If $S = \{k_1, k_2, \cdots, k_{|S|}\} \subset Z_{1,N}$, and $k_1 < k_2 < \cdots < k_{|S|}$, define $(x.)_S = (x_{k_1}, x_{k_2}, \cdots, x_{k_{|S|}})$ and $(\underline{x}.)_S = (\underline{x}_{k_1}, \underline{x}_{k_2}, \cdots, \underline{x}_{k_{|S|}})$. Sometimes, we also abbreviate $(x.)_{Z_{1,N}}$ (i.e., the vector that includes all the possible x_j components) by just $x.$, and $(\underline{x}.)_{Z_{1,N}}$ by just $\underline{x}.$.

Let Z_{ext} be the set of all $j \in Z_{1,N}$ such that $\underline{x}_j$ is an external node, and let Z_{int} be the set of all $j \in Z_{1,N}$ such that $\underline{x}_j$ is an internal node. Clearly, Z_{ext} and Z_{int} are disjoint and their union is $Z_{1,N}$.

Each possible value $x.$ of $\underline{x}.$ defines a different *net story.* For any net story $x.$, we call $(x.)_{Z_{int}}$ the *internal state of the story* and $(x.)_{Z_{ext}}$ its *external state.*

For each net story, we may assign an amplitude to each node. Define S_j to be the set of all k such that an arrow labelled x_k (i.e., an arrow whose source node is $\underline{x}_k$) enters node $\underline{x}_j$. We assign a complex number $A_j[x_j|(x.)_{S_j}]$ to node $\underline{x}_j$. We call $A_j[x_j|(x.)_{S_j}]$ the *amplitude of node* $\underline{x}_j$ *within net story* $x.$.

The *amplitude of net story* $x.$, call it $A(x.)$, is defined to be the product of all the node amplitudes $A_j[x_j|(x.)_{S_j}]$ for $j \in Z_{1,N}$. Thus,

$$A(x.) = \prod_{j \in Z_{1,N}} A_j[x_j|(x.)_{S_j}] \ . \tag{1}$$

The function A_j with values $A_j[x_j|(x.)_{S_j}]$ determines a matrix that we call the *node matrix of node* $\underline{x}_j$. x_j is the matrix's *row index* and $(x.)_{S_j}$ is its *column index*.

Method

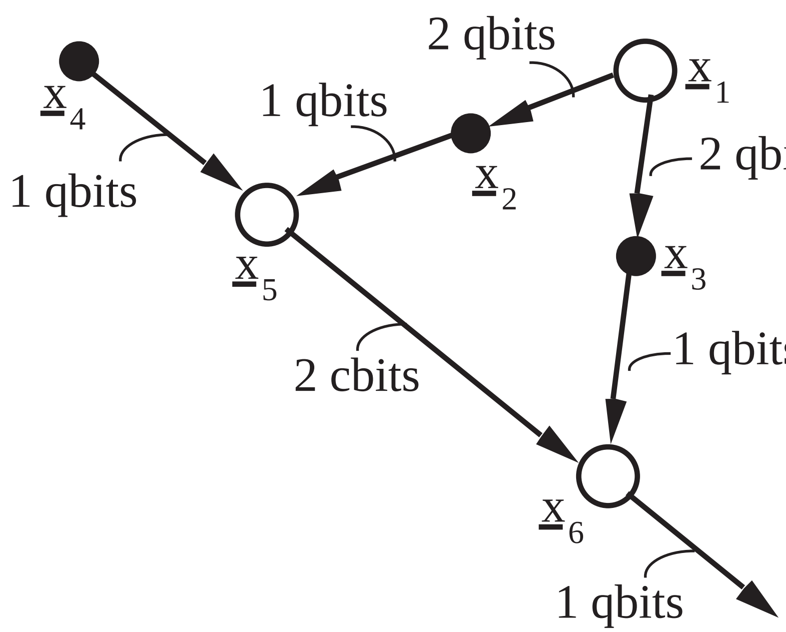

Fig.1 QB net for Teleportation. This figure also shows the number of quantum or classical bits carried by each arrow.

One can translate a QB net into a SEO by performing the following 3 steps: (1) Find eras, (2) Insert delta functions, (3) Find unitary extensions of era matrices. Next we will discuss these steps in detail. We will illustrate our discussion by using Teleportation [10] as an example. Figure 1 shows a QB net for Teleporation. This net is discussed in Ref.[3]. Reference [11] gives a SEO, expressed graphically as a qubit circuit, for Teleportation. It appears that the author of Ref.[11] obtained his circuit mostly by hand, based on information very similar to that contained in a QB net. This paper gives a general method whereby such circuits can be obtained from a QB net in a completely mechanical way by means of a classical computer.

Step 1: Find eras

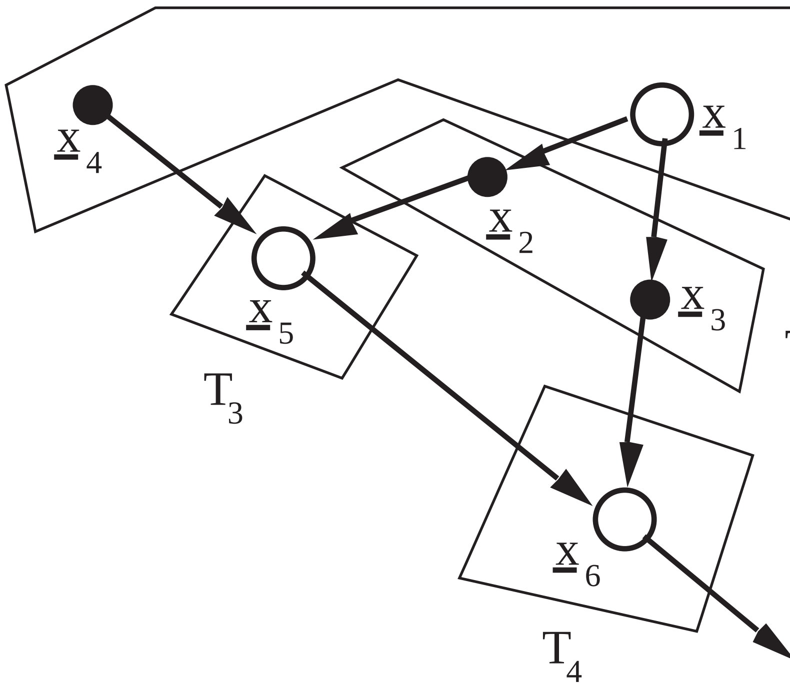

Fig.2 Root node eras for Teleportation net.

The *root node eras* of a graph are defined as follows. Call the original graph Graph(1). The first era T_1 is defined as the set of all root nodes of Graph(1). Call Graph(2) the graph obtained by erasing from Graph(1) all the T_1 nodes and any arrows connected to these nodes. Then T_2 is defined as the set of all root nodes of Graph(2). One can continue this process until one defines an era $T_{|\mathcal{T}|}$ such that Graph($|\mathcal{T}|+1$) is empty. (One can show that if Graph(1) is acyclic, then one always arrives at a Graph($|\mathcal{T}|+1$) that is empty.) For example, Fig.2 shows the root node eras for the Teleportation net Fig.1. Let $\mathcal{T}$ represent the set of eras: $\mathcal{T} = \{T_1, T_2, \cdots, T_{|\mathcal{T}|}\}$. Note that $T_a \subset Z_{1,N}$ for all $a \in Z_{1,|\mathcal{T}|}$ and the union of all T_a equals $Z_{1,N}$. In mathematical parlance, the collection of eras is a partition of $Z_{1,N}$.

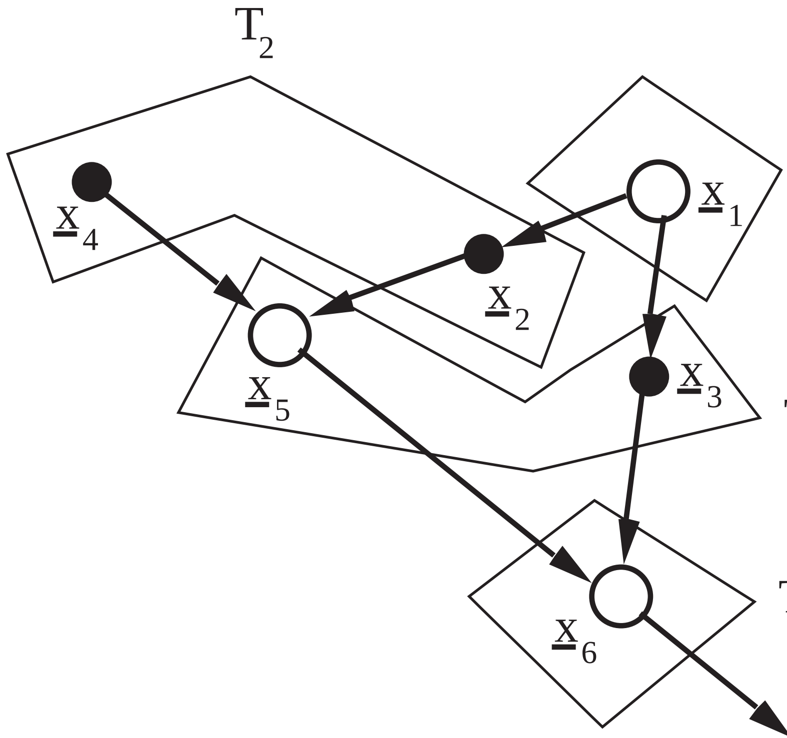

Fig.3 External node eras for Teleportation net.

Rather than defining eras by (1) removing successive layers of root nodes, one can also define them by (2) removing successive layers of external nodes. We call this second type of era, the *external node eras* of the graph. For example, Fig.3 shows the external node eras of the Teleportation net Fig.1.

This process whereby one classifies the nodes of an acyclic graph into eras is a well know technique referred to as a chronological or topological sort in the computer literature[12].

Henceforth, for the sake of definiteness, we will speak only of root node eras. The case of external node eras can be treated similarly.

Suppose that $a \in Z_{1,|\mathcal{T}|}$. The arrows exiting the a'th era are labelled by $(x.)_{T_a}$. Those entering it are labelled by $(x.)_{\Gamma_a}$, where Γ_a is defined by $\Gamma_a = \bigcup_{j \in T_a} S_j$. Note that the a'th era node is only entered by arrows from nodes that belong to previous (not subsequent) eras so $\Gamma_a \subset T_{a-1} \cup \ldots \cup T_2 \cup T_1$. The amplitude B_a of the a'th era is defined as

$$B_a[(x.)_{T_a}|(x.)_{\Gamma_a}] = \prod_{j \in T_a} A_j[x_j|(x.)_{S_j}] \ . \tag{2}$$

The amplitude $A(x.)$ of story $x.$ is given by

$$A(x.) = \prod_{a=1}^{|\mathcal{T}|} B_a \ . \tag{3}$$

For example, for Teleportation we get from Fig.2

$$B_1(x_1, x_4) = A_1(x_1)A_4(x_4) \ , \tag{4a}$$

$$B_2(x_2, x_3|x_1) = A_2(x_2|x_1)A_3(x_3|x_1) \ , \tag{4b}$$

$$B_3(x_5|x_2, x_4) = A_5(x_5|x_2, x_4) \ , \tag{4c}$$

$$B_4(x_6|x_3, x_5) = A_6(x_6|x_3, x_5) \ , \tag{4d}$$

and

$$A(x.) = B_4B_3B_2B_1 \ . \tag{5}$$

Step 2: Insert delta functions

The Feynman Integral FI for a QB net is defined by

$$FI[(x.)_{Z_{ext}}] = \sum_{(x.)_{Z_{int}}} A(x.) \ . \tag{6}$$

Note that we are summing over all stories $x.$ that have $(x.)_{Z_{ext}}$ as their external state. We want to express the right side of Eq.(6) as a product of matrices.

Consider how to do this for Teleportation. In that case one has

$$FI(x_6) = \sum_{x_1, x_2, \ldots x_5} B_4B_3B_2B_1 \ , \tag{7}$$

where the B_a are given by Eqs(4). The right side of Eq.(7) is not ready to be expressed as a product of matrices because the column indices of B_{a+1} and the row indices of B_a are not the same for all $a \in Z_{1,|\mathcal{T}|-1}$. Furthermore, the variable x_3 occurs in B_4 and B_2 but not in B_3. Likewise, the variable x_4 occurs in B_3 and B_1 but not in B_2. Suppose we define $\overline{B}_a$ for $a \in Z_{1,|\mathcal{T}|}$ by

$$\overline{B}_1(x_1^1, x_4^1) = B_1(x_1^1, x_4^1) \ , \tag{8a}$$

$$\overline{B}_2(x_2^2, x_3^2, x_4^2|x_1^1, x_4^1) = B_2(x_2^2, x_3^2|x_1^1)\delta(x_4^2, x_4^1) \ , \tag{8b}$$

$$\overline{B}_3(x_3^3, x_5^3|x_2^2, x_3^2, x_4^2) = B_3(x_5^3|x_2^2, x_4^2)\delta(x_3^3, x_3^2) \ , \tag{8c}$$

$$\overline{B}_4(x_6|x_3^3, x_5^3) = B_4(x_6|x_3^3, x_5^3) \ . \tag{8d}$$

Then

$$FI(x_6) = \sum_{interm} \overline{B}_4 \overline{B}_3 \overline{B}_2 \overline{B}_1 \ , \tag{9}$$

where we sum over all intermediate indices; i.e., all x_j^a except x_6. Contrary to Eq.(7), the right side of Eq.(9) can be expressed immediately as a product of matrices since now B_{a+1} column indices and B_a row indices are the same. The purpose of inserting a delta function of x_3 into B_3 is to allow the system to "remember" the value of x_3 between non-consecutive eras T_4 and T_2. Inserting a delta function of x_4 into B_2 serves a similar purpose.

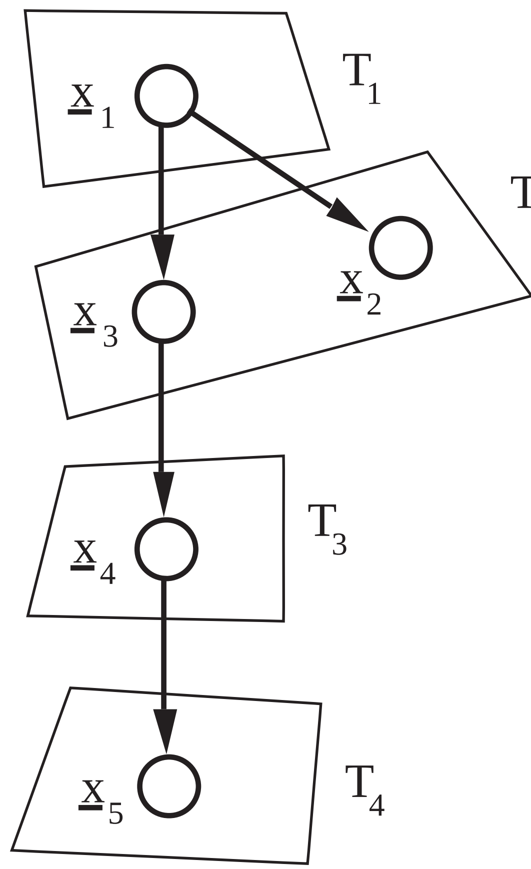

Fig.4 Example of a QB net in which an external node is not in the final era.

In the Teleporation net of Fig.1, the last era contains all the external nodes. However, for some QB nets like the one in Fig.4, this is not the case. For the net of Fig.4,

$$B_1(x_1) = A_1(x_1) \ , \tag{10a}$$

$$B_2(x_2, x_3|x_1) = A_3(x_3|x_1)A_2(x_2|x_1) \ , \tag{10b}$$

$$B_3(x_4|x_3) = A_4(x_4|x_3) \ , \tag{10c}$$

$$B_4(x_5|x_4) = A_5(x_5|x_4) \ . \tag{10d}$$

Even though node $\underline{x}_2$ is external, the variable x_2 does not appear as a row index in B_4. Suppose we set

$$\overline{B}_1(x_1^1) = B_1(x_1^1) \ , \tag{11a}$$

$$\overline{B}_2(x_2^2, x_3^2|x_1^1) = B_2(x_2^2, x_3^2|x_1^1) \ , \tag{11b}$$

$$\overline{B}_3(x_2^3, x_4^3|x_2^2, x_3^2,) = B_3(x_4^3|x_3^2)\delta(x_2^3, x_2^2) \ , \tag{11c}$$

$$\overline{B}_4(x_2, x_5|x_2^3, x_4^3) = B_4(x_5|x_4)\delta(x_2, x_2^3) \ . \tag{11d}$$

Then

$$FI(x_2, x_5) = \sum_{interm} \overline{B}_4\overline{B}_3\overline{B}_2\overline{B}_1 \ , \tag{12}$$

where we sum over all intermediate indices; i.e., all x_j^a except x_2 and x_5. Contrary to B_4, the rows of $\overline{B}_4$ are labelled by the indices of both external nodes $\underline{x}_2$ and $\underline{x}_5$.

This technique of inserting delta functions can be generalized as follows to deal with arbitrary QB nets. For $j \in Z_{1,N}$, let $a_{min}(j)$ be the smallest $a \in Z_{1,|\mathcal{T}|}$ such that x_j appears in B_a. Hence, $a_{min}(j)$ is the first era in which x_j appears. If $\underline{x}_j$ is an internal node, let $a_{max}(j)$ be the largest a such that x_j appears in B_a (i.e., the last era in which x_j appears). If $\underline{x}_j$ is an external node, let $a_{max}(j) = |\mathcal{T}| + 1$. For $a \in Z_{1,|\mathcal{T}|}$, let

$$\Delta_a = \{j \in Z_{1,N} | a_{min}(j) < a < a_{max}(j)\} \ , \tag{13}$$

$$\overline{B}_a = B_a[(x_\cdot^a)_{T_a}|(x_\cdot^{a-1})_{\Gamma_a}] \prod_{j\in\Delta_a} \delta(x_j^a, x_j^{a-1}) \ . \tag{14}$$

In Eq.(14), $x_j^{|\mathcal{T}|}$ should be identified with x_j and x_j^0 with no variable at all. Equation(6) for FI can be written in terms of the $\overline{B}_a$ functions:

$$FI[(x_\cdot)_{Z_{ext}}] = \sum_{interm} \overline{B}_{|\mathcal{T}|} \ldots \overline{B}_2\overline{B}_1 \ , \tag{15}$$

where the sum is over all intermediate indices (i.e., all x_j^a for which $a \neq |\mathcal{T}|$). For all a, define matrix M_a so that the x, y entry of M_a is $\overline{B}_a(x|y)$. Define M to be a column

vector whose components are the values of FI for each external state. Then Eq.(15) can be expressed as:

$$M = M_{|\mathcal{T}|} \dots M_2 M_1 \ . \tag{16}$$

The rows of the column vector M are labelled by the possible values of $(x.)_{Z_{ext}}$. The rows of the column vector M_1 are labelled by the possible values of $(x.)_{T_1}$, where T_1 is the set of root nodes.

Step 3: Find unitary extensions of era matrices

So far, we have succeeded in expressing FI as a product of matrices M_a, but these matrices are not necessarily unitary. In this step, we will show how to extend each M_a matrix (by adding rows and columns) into a unitary matrix U_a. The techniques of Ref.[4] will then be applicable to each matrix U_a.

By combining adjacent M_a's, one can produce a new, smaller set of matrices M_a. Suppose the union of two consecutive eras is also defined to be an era. Then combining adjacent M_a's is equivalent to combining two consecutive eras to produce a new, smaller set of eras. We define a *breakpoint* as any position $a \in Z_{1,|\mathcal{T}|-1}$ between two adjacent matrices M_{a+1} and M_a. Combining two adjacent M_a's eliminates a breakpoint. Breakpoints are only necessary at positions where internal measurements are made. For example, in Teleportation experiments, one measures node $\underline{x}_3$, which is in era T_3. Hence, a breakpoint between M_4 and M_3 is necessary. If that is the only internal measurement to be made, all other breakpoints can be dispensed with. Then we will have $M = M_2' M_1'$ where $M_2' = M_4$, $M_1' = M_3 M_2 M_1$. If no internal measurements are made, then we can combine all matrices M_a into a single one, and eliminate all breakpoints.

We will henceforth assume that for all $a \in Z_{1,|\mathcal{T}|}$, the columns of M_a are orthonormal. If for some $a_0 \in Z_{1,|\mathcal{T}|}$, M_{a_0} does not satify this condition, it may be possible to "repair" M_{a_0} so that it does. First: If a row β of M_{a_0-1} is zero, then eliminate the column β of M_{a_0}, and the row β of M_{a_0-1}. Next: If a row β of the column vector $M_{a_0-1} \dots M_2 M_1$ is zero, then flag the column β of M_{a_0}. The flagged columns of M_{a_0} can be changed without affecting the value of M. If the non-flagged columns of M_{a_0} are orthonormal, and the number of columns in M_{a_0} does not exceed the number of rows, then the Gram Schmidt method, to be discussed later, can be used to replace the flagged columns by new columns such that all the columns of the new matrix M_{a_0} are orthonormal. If it is not possible to repair M_{a_0} in any of the above ways (or in some other way that might become clear once we program this), one can always remove the breakpoint between M_{a_0+1} and M_{a_0}.

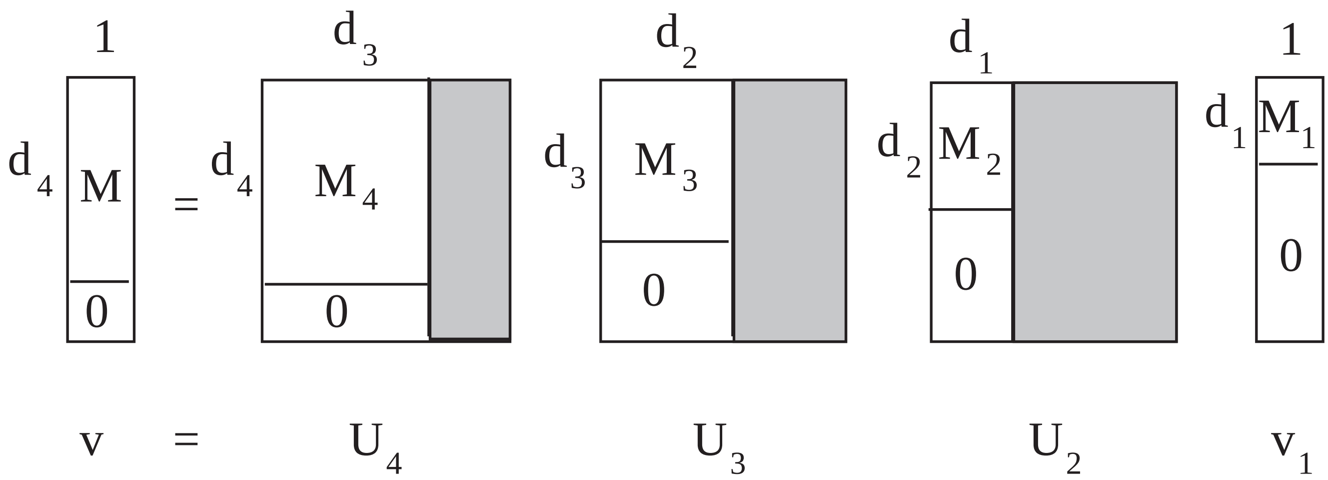

Fig.5 Dimensions of matrices M_a and of their unitary extensions U_a.

Consider Fig.5. We will call d_a the number of rows of matrix M_a and d_{a-1} its number of columns. We define D and N_S by

$$D = \max\{d_a | 1 \leq a \leq |\mathcal{T}|\} \tag{17}$$

$$N_S = \min\{2^i | i \in Z_{1,\infty}, D \leq 2^i\} . \tag{18}$$

Let $\overline{d}_a = N_S - d_a$ for all a.For each $a \neq 1$, we define U_a to be the matrix that one obtains by extending M_a as follows. We append an $\overline{d}_a \times d_{a-1}$ block of zeros beneath M_a and an $N_S \times \overline{d}_{a-1}$ block of gray entries to the right of M_a. By gray entries we mean entries whose value is yet to be specified. When $a = 1$, M_1 be can extended in two ways. One can append a column of $\overline{d}_1$ zeros beneath it and call the resulting N_S dimensional column vector v_1. Alterntatively, one can append a column of $\overline{d}_1$ zeros beneath M_1 and an $N_S \times (N_S - 1)$ block of gray entries to the right of M_1, and call the resulting $N_S \times N_S$ matrix U_1. In this second case, one must also insert e_1 to the right of U_1. By e_1 we mean the N_S dimensional column vector whose first entry equals one and all others equal zero. Which extension of M_1 is used, whether the one that requires e_1 or the one that doesn't, should be left as a choice of the user. Henceforth, for the sake of definiteness, we will will assume that the user has chosen the extension without the e_1. The other case can be treated similarly. Equation(16) then becomes

$$v = U_{|\mathcal{T}|} \ldots U_3 U_2 v_1 , \tag{19}$$

where v is just the column vector M with $\overline{d}_{|\mathcal{T}|}$ zeros attached to the end. Note that

$$U_a \ldots U_2 v_1 = \begin{bmatrix} M_a \ldots M_2 M_1 \\ 0 \end{bmatrix} , \tag{20}$$

for all $a \in Z_{1,|\mathcal{T}|}$, where the zero indicates a column of $\overline{d}_a$ zeros.

To determine suitable values for the gray entries of the U_a matrices, one can use the Gram-Schmidt (G.S.) method [13]. This method takes as input an ordered set $S = (v_1, v_2, \ldots, v_N)$ of vectors, not necessarily independent ones. It yields as output another ordered set of vectors $S' = (u_1, u_2, \ldots, u_N)$, such that S' spans the same vector space as S. Some vectors in S' may be zero. Those vectors of S' which aren't zero will be orthonormal. For $r \in Z_{1,N}$, if the first r vectors of S are already orthonormal, then the first r vectors of S' will be the same as the first r vectors of S. Let e_j for $j \in Z_{1,N_S}$ be the j'th standard unit vector (i.e., the vector whose j'th entry is one and all other entries are zero). For each $a \in Z_{1,|\mathcal{T}|}$, to determine the gray entries of U_a one can use the G.S. method on the set S consisting of the non-gray columns of U_a together with the vectors $e_1, e_2, \ldots e_{N_S}$.

References

[1] R.R. Tucci, Int. Jour. of Mod. Physics **B9**, 295 (1995). Los Alamos eprint quant-ph/9706039

[2] R.R. Tucci, "Graphical Computer Method for Analyzing Quantum Systems", US Patent Pending.

[3] A commercial software program called "Quantum Fog" that simulates QB nets is available at www.ar-tiste.com

[4] R.R. Tucci, "A Rudimentary Quantum Compiler", Los Alamos eprint quant-ph

[5] D.P. DiVincenzo, Science **270**, 255 (1995).

[6] P. Shor, SIAM Journal of Computing **26**, 1484 (1997). Available as Los Alamos eprint quant-ph/9508027.

[7] P. W. Shor, Phys. Rev. A **52**, 2493 (1995); R. Laflamme, C. Miquel, J. P. Paz, and W. H. Zurek, Phys. Rev. Let. **77**, 198 (1996); A. M. Steane, Phys. Rev. A **54**, 4741 (1996); A. M. Steane, Phys. Rev. Let. **77**, 793 (1996).

[8] Ari Kornfeld, AI Expert, Nov. 1991, pg.42; Eugene Charniak, AI Magazine, Winter 1991, pg. 50; Max Henrion, John S. Breese, Eric J. Horowitz, AI Magazine, Winter 1991, pg. 64; S.L. Lauritzen, D.J.Spiegelhalter, J. Roy. Stat. Soc. B **50**, pgs. 157-224 (1988); S.K. Andersen, K.G. Olessen, F.V. Jensen, F. Jensen, Proc. of IJCAI **89**, pgs. 1080-1085 (1989); Richard E. Neapolitan, *Probabilistic Reasoning in Expert Systems: Theory and Algorithms* (Wiley, 1990); Judea Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference* (Morgan Kaufmann, Palo Alto, 1988);

[9] Leslie Helm, “Improbable Inspiration”, LA Times, Oct. 28, 1996.

[10] C.H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, W.K. Wooters, Physical Review Letters **70**, 1895 (1993).

[11] G. Brassard, Los Alamos eprint quant-ph/9605035.

[12] Bryan Flamig, *Practical Algorithms in C++* (Wiley, 1995) page 369.

[13] B. Noble and J.W. Daniels, *Applied Linear Algebra*, Third Edition (Prentice Hall, 1988).

FIGURE CAPTIONS:

FIG.1 QB net for Teleportation. This figure also shows the number of quantum or classical bits carried by each arrow.

FIG.2 Root node eras for Teleportation net.

FIG.3 External node eras for Teleportation net.

FIG.4 Example of a QB net in which an external node is not in the final era.

FIG.5 Dimensions of matrices M_a and of their unitary extensions U_a.